A fast pivot-based indexing algorithm for metric spaces

Raisa Socorro, Luisa Micó, and Jose Oncina

[1] Instituto Superior Politécnico José Antonio Echevarría (La Habana, Cuba
[2] Dept. Lenguajes y Sistemas Informáticos (Universidad de Alicante, E-03071 Alicante, Spain)

Abstract. This work focus on fast Nearest Neighbour (NN) search algorithms that can work in any metric space (not just the Euclidean distance) and where the distance computation is very time consuming. One of the most well known methods in this field is the AESA algorithm, used as baseline for performance measurement for over twenty years. The AESA works in two steps that repeats: first it searches a promising candidate to NN and computes its distance (approximation step), next it eliminates all the unsuitable NN candidates in view of the new information acquired in the previous calculation (elimination step). This work introduces the PiAESA algorithm. This algorithm improves the performance of the AESA algorithm by splitting the approximation criterion: on the first iterations, when there is not enough information to find good NN candidates, it uses a list of pivots (objects in the database) to obtain a cheap approximation of the distance function. Once a good approximation is obtained it switches to the AESA usual behaviour. As the pivot list is built in preprocessing time, the run time of PiAESA is almost the same than the AESA one.
In this work, we report experiments comparing with some competing methods. Our empirical results show that this new approach obtains a significant reduction of distance computations with no execution time penalty.

1 Introduction

Methods based on similarity search [30] (also called Nearest Neighbour (NN) search when the dissimilarity measure is interpreted as a distance) are having an increasing interest due to its simplicity and the emergence of new research fields. Some examples are Text and Image Retrieval [23][19][3], Relevance Feedback for Content Based Image Retrieval [12][29], Multimedia Databases[24][14], Pattern Recognition[1][22], Stream Mining [25], etc.

In order to quickly find the most similar object in a database to a given query object, the goal of many fast search algorithms is reducing, due to its usual high cost, the number of similarity computations by avoiding a search throughout the full database. In order to achieve such objective, some type of restricting property, that the similarity measure should meet, is exploited. The triangular

inequality is the most common one since there is a large number of similarity measures that fulfils it: Euclidean distance, Manhattan distance [16], or the more general Minkowski distance [17], edit distance [18], quadratic form distances, Hausdorff distance, Bhattacharyya distance, etc. Since some of these distances are widely used (i.e. Euclidean distance) there are specialised fast NN search algorithms for them [11]. In this work we focus on fast NN search algorithms that can deal with *any* similarity measure provided it defines a metric in the space of the objects stored in the database. Some examples of the usefulness of these fast techniques can be found in the literature [15][27][21][3].

All fast NN search algorithms build a data structure (index), at preprocessing time, that is used to speed up the search. Along the years, several taxonomies have been defined to classify the search algorithms depending on the space partitioning criteria used to build the index or the type of transformation strategies used [7], [13]. According to the taxonomy proposed by Chávez *et al.* [7], the Approximating and Eliminating Search Algorithm (AESA), introduced by E. Vidal in 1986 [26], is classified as a pivot-based metric space search algorithm. The pivots are a subset of objects in the database that are used to speed up the search. Usually, the distances from each pivot to some (or all) of the rest of the objects in the database are stored in preprocessing time and used during the search to avoid distance computations.

For two decades [8], AESA is being considered the fastest NN search methods in metric spaces (measured in number of distance computations). In order to find the NN to a target object, this algorithm works by iterating two strategies: first, it searches for (heuristically) a candidate to NN (the approximating step) and second, it uses this candidate to discard all the objects in the database than can not be nearest to the target than the current candidate (the elimination step).

Focusing on reducing the number of distance computations Figueroa *et al.* [8] proposed the iAESA algorithm. This algorithm is based on the use of a different approximating step: each candidate stores a list of the previously used pivots sorted by closeness to it, and chooses the next pivot as the candidate, whose list (a pivot permutation) is most similar to that of the query. In this approximation, the overhead produced in the search is much higher than in the AESA algorithm because it must maintain the permutations updated.

The main idea of our approach is to realize that AESA, in the first iterations, has little information about the location of the target, and then, it can not propose good candidates to NN. To exploit this, PiAESA proposes, during the first iterations, pivots oriented to increase the accuracy of the cheap (constant computation time) alternative dissimilarity function that is used by AESA to approximate the real one. When the alternative dissimilarity function is accurate enough PiAESA switches to the usual strategy of AESA. The list of pivots to propose is computed in preprocessing time and then there is not execution time penalty with respect to AESA. Moreover, many techniques to sort the pivots can be used. Several of them are described and explored in this work.

Experiments using artificial and real data confirm that significant improvements in performance are obtained.

2 The PiAESA

Given a target object q and a database T, in each iteration, AESA searches for a good candidate to NN avoiding to compute as many distances as possible. Instead of searching the object $t \in T$ that minimises the distance by computing all the distances $d(q,t)$ to the target(*exhaustive search*), it uses a lower bound of the distance function: $G(q,t) = \max_{p \in P} |d(q,p) - d(p,t)|$ [1], where P is the set of the NN candidates obtained in the previous iterations. This bound can be easily derived from the triangular inequality taking into account that $d(q,t) \geq |d(q,p) - d(p,t)| \ \forall p,q,t$.

Each time a new NN candidate p is obtained, the distance $d(q,p)$ to the target is computed and the lower bound is updated. At preprocessing time all the distances $d(r,s), \forall r,s \in T$ are computed and stored in the index. Then, as the objects p and t belong to the database, the distance $d(p,t)$ can be obtained in constant time and the bound can be updated without computing any additional distance.

In AESA, this bound is used for both, approximating and eliminating. For the approximation step AESA finds the object that minimizes the lower bound distance, and the elimination step AESA removes the objects whose lower bound distance to the target is bigger than the present distance from the NN candidate to the target.

Note that in the first iterations, since there are few objects in P, the lower bound distance is a very poor estimation of the actual distance, and the NN candidates will be inadequate. In our approach, instead of insisting on finding the best candidate to NN, in the first stages of the algorithm we find the objects that will increase most the value of the lower bound making it a good estimation on the distance. In order to assess the accuracy of the lower bound, a parameter R is used. If during R iterations the distance to the current NN does not change, we consider the lower bound is accurate enough and we switch to the AESA strategy (see algorithm 1). Cross validation techniques can be used to fix a suitable value for R.

In our approach the objects in the database are sorted, at preprocessing time, according to its expected contribution to the grow of the lower bound. In the next section we are going to review some competing methods of sorting database objects. Most of them are borrowed from some other fast NN search algorithms where a similar problem appears.

The resulting algorithm is an extension of AESA guided by the parameter R. Note that when $R = 0$ the algorithm is exactly the AESA.

3 Pivot Selection Techniques

As it has been mentioned above, several fast search algorithms have faced the problem of selecting and sorting the objects of the database in order to reduce the

[1] Note that in algorithm 1 the lower bound is written as $G(t)$ since q, the target object, does not change in the algorithm.

Algorithm 1: PiAESA

Input:
T: training set
q: query
$P \subset T$: list of ordered pivots
$D \in \mathbb{R}^{|T| \times |T|}$: table of distances
$R \in \mathbb{N}$: parameter to control the change of approximation criterion

Output:
$p_{min} \in T$: nearest neighbour to q

```
1  d_min = ∞                                                    // initialisation
2  foreach t ∈ T do G(t) = 0
3  i = 0; g_min = 0; g_prev_min = 0
4  while P ≠ ∅ and i < R do
5  |  s = extract_first(P); T = T − {s}
6  |  d = d(q, s)
7  |  if d < d_min then p_min = s; d_min = d                    // new NN
8  |  g_prev_min = g_min; g_min = ∞
9  |  foreach t ∈ T do
10 |  |  G(t) = max(G(t), |D(t, s) − d|)
11 |  |  if G(t) < g_min then g_min = G(t)
12 |  end
13 |  i = i + 1
14 |  if g_min > g_prev_min then i = 0
15 end
16 while T ≠ ∅ do
17 |  s = argmin_{u∈T} G(u)                                     // approximation
18 |  T = T − {s}
19 |  d = d(q, s)
20 |  if d < d_min then p_min = s; d_min = d                    // new NN
21 |  foreach t ∈ T do
22 |  |  G(t) = max(G(t), |D(t, s) − d|)
23 |  |  if G(t) ≥ d_min then T = T − {t}                       // elimination
24 |  end
25 end
```

number of comparisons. Those technique are known as *pivot selecting techniques* [2][9][20]. Let us review the ones we are going to use in our algorithm.

3.1 Random Pivot Selection *(RPS)*

In this naïve approach, pivots are selected randomly. In our case we are going to force that all the objects in the database appear in the pivot list, then this method becomes a random enumeration of the objects in the database.

This technique was included here as a base line to be able to quantify the contribution of the other techniques. Surprisingly, this technique obtain quite good results in our real data experiments.

3.2 Incremental Outlier Selection Techniques

These techniques select incrementally objects located far away from the previously selected objects. Starting with a randomly selected pivot (p_1), two strategies are used to select the next pivot [20].

Maximum of Minimum Distances (MMD)

$$p_i = \underset{s \in T - \{p_1, \ldots, p_{i-1}\}}{\operatorname{argmax}} \min_{j=1}^{i-1} d(s, p_j)$$

Maximum of Sum of Distances (MSD)

$$p_i = \underset{s \in T - \{p_1, \ldots, p_{i-1}\}}{\operatorname{argmax}} \sum_{j=1}^{i-1} d(s, p_j)$$

The list of pivots is then, $P = (p_1, \ldots, p_{|T|})$.

Note that, as in the random pivot selection technique (RPS), all the objects in the database appears in the pivot list.

These techniques are usually applied in LAESA [20] algorithm, but, in this algorithm, the listing is pruned depending on a parameter.

3.3 Sparse Spatial Selection (SSS)

This method dynamically selects a set of pivots uniformly distributed in the space, and adapted to the complexity of the database [5][4].

In this technique the set of pivots contains initially only a randomly chosen object of the database. The remaining objects are randomly considered for the inclusion in the pivot set. An object is included in the pivot list if its distance to any already selected pivot is greater than or equal to a fraction (α) of the maximum distance between objects in the database (M). This restriction ensures that all pivots are well distributed in the whole space.

The parameter α is usually chosen in the interval $[0.35, 0.40]$. In this method, pivots are not very far away from each others neither very far from the rest of objects in the database, but they are well distributed covering the whole space. This method was used in a basic proximity search pivot-based algorithm described in [5]. In this work, the value of α was obtained experimentally, obtaining the better results for α=0.40.

Note that two objects at a distance lower that αM can not be at the same time in the pivot list. As a consequence, the pivot list is not usually an enumeration of the objects in the database. PiAESA has a special condition to force switching to AESA behaviour when the pivot list is empty. Note that this case is undesirable since we know that the lower bound is not accurate enough.

3.4 Dynamic Pivot Selection (DPS)

This technique, proposed in [6], is a dynamic extension of the SSS method and, like the previous method, was used in the same search algorithm. The method uses an efficiency criterion proposed in [5] to determine the contribution of each new candidate to pivot. As in the SSS case, no candidate is considered if there is a pivot at a distance lower than αM. If the number of pivots already selected is smaller than a threshold, the candidate is added to the pivot list. Otherwise, the method uses the efficiency criterion to determine the contribution of the candidate. Depending on that, the candidate is discarded, added to the pivot list or replaces the current worst pivot.

Similar to the previous case, the list of pivots is not an enumeration of the objects in the database and we have to allow, as in the SSS case, switching to AESA behaviour when the list becomes empty.

4 Experimental Results

In this work we have carried out a series of experiments using artificial and real data to check the performance of our proposal.

Three sets of databases were used in our experiments:

1. A synthetic set of databases generated by drawing uniformly points from the unit hypercube from dimension $k = 2$ to 24, obtained from `http://www.sisap.org`[2]. In these experiments Minkowski L_1 distance (*Manhattan distance*) was used as dissimilarity measure. Every k-dimensional vector space has been used as a metric space and, then, the points were treated as abstract objects in an unknown metric space.
2. Two real data databases: NASA and COLOR, obtained from `http://www.sisap.org`. Both databases are feature vectors extracted from two collections of images: NASA contains 40150 20-dimensional vectors obtained from

[2] Oficial Website of the International Workshop on Similarity Search and Applications, with source code, benchmarks and bibliography for the similarity search community

NASA video and image archives. COLORS collection contains 112682 112-dimensional vector obtained from color images. As well as in the previous case, the Minkowski L_1 distance was used as dissimilarity measure.

3. A string database representing contour chains [10] of the digits in MNIST database (`http://yann.lecun.com/exdb/mnist`). In this case the edit distance [18][28] was used as dissimilarity measure.

4.1 Analysis of the parameter R and the pivot selection techniques

PiAESA has two parameters, an integer parameter R, and a functional parameter: the pivot selection technique. This section is devoted to study the influence of both parameters in the PiAESA behaviour.

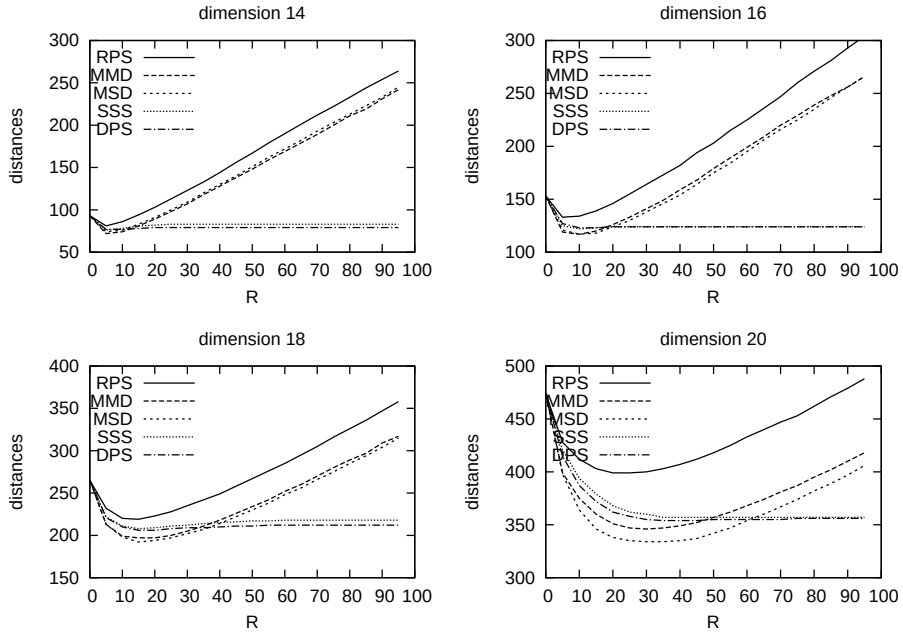

Fig. 1. Variation of the average number of distance computations when increasing the parameter R for 14,16,18 and 20 dimensions unit hypercube data. Training and test data size were 15 000 and 1 000 respectively.

Figures 1, 2 and 3 shows the evolution of the number of distance computations as the parameter R increases, for the selection techniques described in section 3 (RPS, MMD, MSD, SSS and DPS) and for the three database sets described at the beginning of the section. All the experiments were made with a 15 000 random subset of the whole database. Each point is the average of 1 000 target objects.

It can be observed that in all the cases, except in the NASA database where the improvement is negligible, there is an optimum value for R. Surprisingly the best pivot selection technique for the handwriting database is the RPS.

Moreover, the synthetic database results suggest that the improvements with respect to the AESA increases with the dimension.

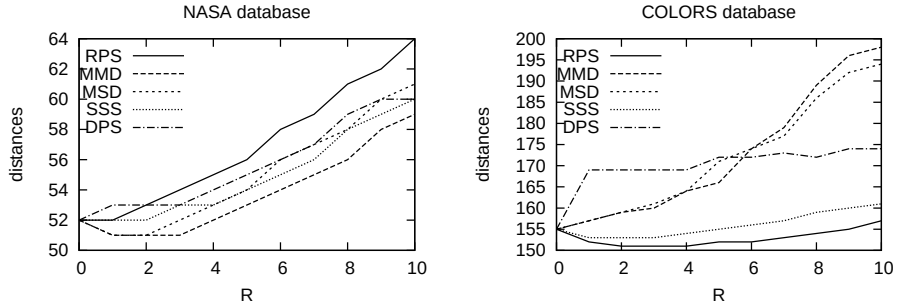

Fig. 2. Variation of the average number of distance computations when increasing the parameter R for NASA (left) and COLORS (right) databases. Training and test data size were 15 000 and 1 000 respectively.

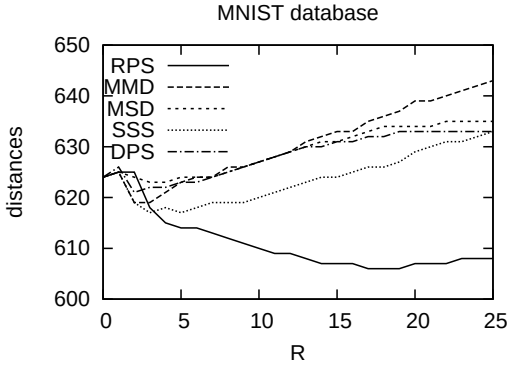

Fig. 3. Average number of distance computations when increasing the parameter R for a training size of 15 000 contour strings handwritten digits.

In PiAESA the number of pivots used before the switching to AESA behaviour is not fixed and depends on the parameter R. We have conducted a series of experiments in order to show this dependency. For a fixed value of R, 15 000 target objects were used over a training set with size 15 000. In each

search, the number of pivots used were counted. The histogram is depicted in figure 4. Although this experiment was done for all the databases and many values of R, in fig. 1, 2 and 3, we show some of them (similar results were obtained with others). It can be observed that very few cases requires a large amount of pivots.

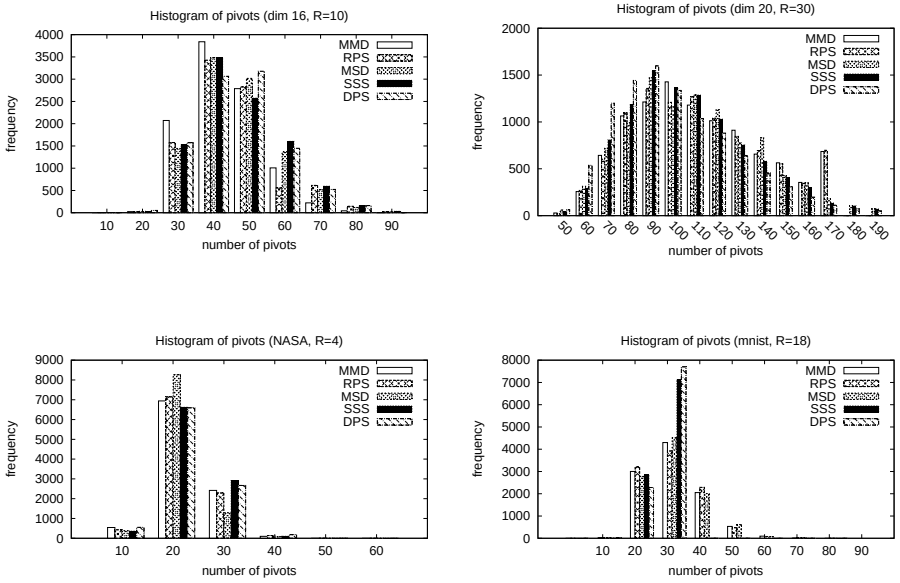

Fig. 4. Histograms of the number of ordered pivots used by the method for a fixed value of the parameter R with artificial data (top) and real data (down). Training and test data sizes were 15 000 objects.

Figure 5 (left) represents the optimum value for R as the dimension increases using the MMD selection technique (synthetic database). It can be observed that the optimum value of R increases very quickly as the dimension grows suggesting that large improvements can be obtained in big dimensionality spaces. Figure 5 (right) shows the variation of the optimum value for R as the database sizes increases from 0 to 15 000. The experiments shows that once a threshold is surpassed, the optimum value for R is independent of the database size.

4.2 Comparison with other methods

In this section we compare the performance of PiAESA using all pivot selection strategies with some competing techniques: AESA, LAESA and iAESA. The results are shown in Table 1 and 2. In nearly all the experiments, PiAESA

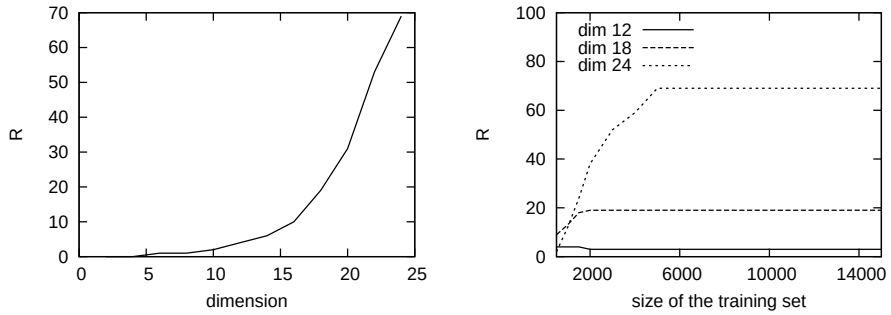

Fig. 5. Performance of the parameter R with different values of the dimensionality of the data (left) and size training set (right).

obtains the best results when either MMD or SMD pivot selection technique is used. The algorithm obtains only better results when RPS pivot technique is used with the database COLORS and MNIST.

Table 1. Average number of distances computed by AESA (synthetic database), LAESA (using 42, 183 and 547 pivots for dimensions 12, 18 and 24 respectively) and PiAESA (the value of R was 3, 19 and 69 for 12, 18 and 24 dimensions). Results for 5000, 10000 and 15000 training sets, 1 000 queries were used for testing.

	Dimension 12			**Dimension 18**			**Dimension 24**		
	5000	**10000**	**15000**	**5000**	**10000**	**15000**	**5000**	**10000**	**15000**
AESA	55.70	55.57	54.32	281.67	289.96	281.26	1067.72	1233.58	1287.70
LAESA	71.26	70.37	68.31	333.73	348.68	346.71	1243.91	1437.56	1543.91
iAESA	51.49	51.15	50.16	248.09	234.25	224.60	991.91	1085.12	1098.19
PiAESA-MMD	**44.96**	**44.49**	**43.57**	211.24	208.67	206.48	889.06	946.37	952.13
PiAESA-SMD	47.96	47.43	46.33	**208.01**	**206.21**	**205.33**	**835.44**	**889.36**	**887.43**
PiAESA-RPS	52.58	52.27	51.43	245.05	240.06	237.78	979.37	1082.22	1090.54
PiAESA-SSS	48.84	47.98	46.55	223.69	219.68	218.73	906.59	991.90	985.07
PiAESA-DPS	47.52	47.52	46.66	227.20	220.96	222.36	909.74	992.15	993.44

The experiments for synthetic data were repeated for a fixed training set with 15 000 samples and dimensions ranging from 2 to 24 (see Figure 6). It can be observed that iAESA algorithm reduces the number of distances computations comparing with AESA as the dimensionality increases, in particular from dimension 10 to 24 (with a reduction of 15%). Comparing the results of PiAESA and AESA, for low dimensions PiAESA does not improve AESA (the best result is obtained with the parameter $R = 0$, which means we are really running the original AESA). When the dimensionality increases and the parameter R takes values greater than 0, the approach outperforms AESA in a 25% and iAESA at 14% for dimension 24.

Table 2. Average number of distances computed by AESA algorithms using a training set of 15000 objects and 1 000 queries with databases NASA and COLORS. For the database NASA, the value of the parameter R for PiAESA was 3 using MMD and 1 using RPS. The value of R with the database COLORS was 0 using MMD and 2 using RPS. The value of R with the database MNIST was 2 using MMD and 18 using RPS.

	NASA	**COLORS**	**MNIST**
AESA	52.31	155.23	624.71
LAESA	111.64	560.36	1736.92
iAESA	51.86	165.44	654.70
PiAESA-MMD	51.54	155.23	619.40
PiAESA-SMD	**51.07**	155.23	617.12
PiAESA-RPS	52.27	**151.35**	**606.53**
PiAESA-SSS	52.31	155.23	623.12
PiAESA-DPS	52.31	155.23	621.61

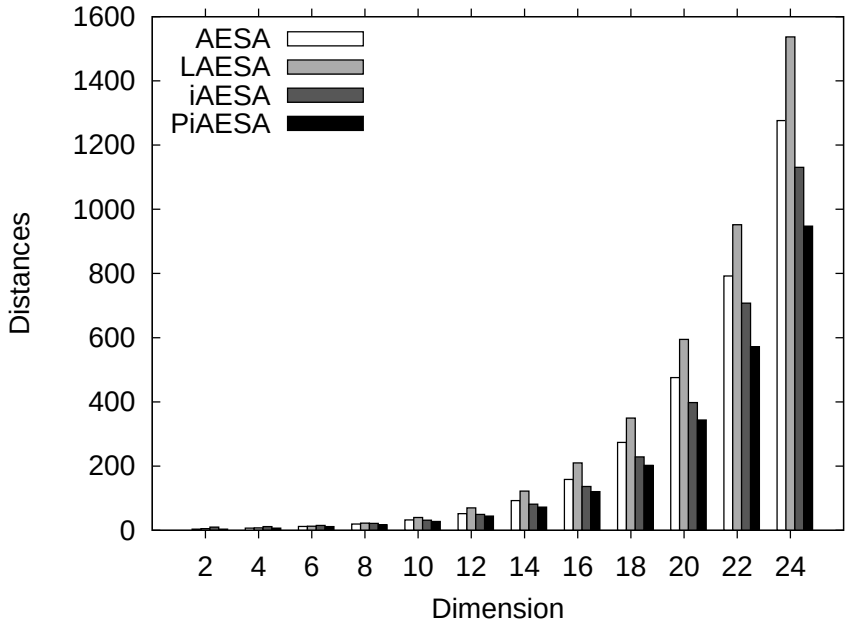

Fig. 6. Average number of distance computations using a training set of 15 000 samples and several dimensions.

5 Conclusions

AESA is a fast nearest neighbour search algorithm in metric spaces that has been for 20 years the baseline method in terms of saved distance computations.

In this work an AESA based new search algorithm has been proposed. The algorithm uses a list of pivots (obtained at preprocessing time) that are sequentially used in the first steps of the algorithm to obtain a cheap good estimation of the distance. After performing that operation it switches to the usual AESA behaviour.

Several methods for selecting pivots were tested in the experiments. It has been shown experimentally that our approach improves both: AESA and other recently proposed algorithms without an extra space or time cost. The improvements are more pronounced in higher dimensions.

As future work, we think that some improvements can be obtained if other pivot selection techniques are used and even new approximation criteria.

Acknowledgements. The authors thank the Spanish CICyT for partial support of this work through projects TIN2009-14205-C04-01, the IST Programme of the European Community, under the PASCAL Network of Excellence, IST–2002-506778, and the program CONSOLIDER INGENIO 2010 (CSD2007-00018).

References

1. M. Ankerst, G. Kastenmüller, H.P. Kriegel, and T. Seidl. 3d shape histograms for similarity search and classification in spatial databases. pages 207–226. Springer, 1999.
2. L. G. Ares, N. Brisaboa, M. F. Esteller, O. Pedreira, and A. S. Places. Optimal pivots to minimize the index size for metric access methods. In *SISAP '09: Proceedings of the 2009 Second International Workshop on Similarity Search and Applications*, pages 74–80, Washington, DC, USA, 2009. IEEE Computer Society.
3. S. Battiato, G. Di Blasi, and D. Reforgiato. Advanced indexing schema for imaging applications: three case studies. *Image Processing, IET*, 1(3):249–268, 2007.
4. N. Brisaboa, A. Farina, O. Pedreira, and N. Reyes. Similarity search using sparse pivots for efficient multimedia information retrieval. In *ISM '06: Proceedings of the Eighth IEEE International Symposium on Multimedia*, pages 881–888, Washington, DC, USA, 2006. IEEE Computer Society.
5. B. Bustos, G. Navarro, and E. Chávez. Pivot selection techniques for proximity searching in metric spaces. *Pattern Recognition Letters*, 24(14):2357–2366, 2003.
6. B. Bustos, O. Pedreira, and N. Brisaboa. A dynamic pivot selection technique for similarity search. In *Proceedings of the First International Workshop on Similarity Search and Applications*, pages 105–112, Washington, DC, USA, 2008. IEEE Computer Society.
7. E. Chávez, G. Navarro, R. Baeza-Yates, and J. L. Marroquín. Searching in metric spaces. *ACM Comput. Surv.*, 33(3):273–321, 2001.
8. K. Figueroa, E. Chávez, G. Navarro, and R. Paredes. Speeding up spatial approximation search in metric spaces. *J. Exp. Algorithmics*, 14:3.6–3.21, 2009.

9. R. F. Filho, A. Traina, C. Traina, Jr., and C. Faloutsos. Similarity search without tears: The omni family of all-purpose access methods. In *Proceedings of the 17th International Conference on Data Engineering*, pages 623–630, Washington, DC, USA, 2001. IEEE Computer Society.
10. H. Freeman. Boundary encoding and processing. In B. Lipkin and A. Rosenfeld, editors, *Picture Processing and Psychopictorics*, pages 241–266. Academic Press, 1970.
11. J.H. Friedman, J.L. Bentley, and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3:209–226, 1977.
12. G. Giacinto. A nearest-neighbor approach to relevance feedback in content based image retrieval. In *CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 456–463, New York, NY, USA, 2007. ACM.
13. G.R. Hjaltason and H. Samet. Index-driven similarity search in metric spaces (survey article). *ACM Trans. Database Syst.*, 28(4):517–580, 2003.
14. T.S. Huang, C.K. Dagli, S. Rajaram, E.Y. Chang, M.I. Mandel, G.E. Poliner, and D.P.W. Ellis. Active learning for interactive multimedia retrieval. volume 96, pages 648–667, 2008.
15. Flip Korn, Nikolaos Sidiropoulos, Christos Faloutsos, Eliot Siegel, and Zenon Protopapas. Fast nearest neighbor search in medical image databases. In *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB '96, pages 215–226, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.
16. E.F. Krause. *Taxicab Geometry: An Adventure in Non-Euclidean Geometry.* New York: Dover, 1986.
17. J.B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a non metric hypothesis. *Psychomotrika*, 29:1–27, 1964.
18. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Doklady Akademii Nauk SSSR*, 163(4):845–848, 1965.
19. Q. Lv, M. Charikar, and K. Li. Image similarity search with compact data structures. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 208–217, New York, NY, USA, 2004. ACM.
20. L. Micó, J. Oncina, and E. Vidal. A new version of the nearest-neighbour approximating and eliminating search algorithm (AESA) with linear preprocessing time and memory requirements. *Pattern Recognition Letters*, 15(1):9–17, 1994.
21. F. Moreno-Seco, L. Micó, and J. Oncina. Approximate nearest neighbour search with the fukunaga and narendra algorithm and its application to chromosome classification. *Lecture Notes in Computer Science - Lecture Notes in Artificial Intelligence*, 2905:322–328, 2003.
22. M. Potamias and V. Athitsos. Nearest neighbor search methods for handshape recognition. In *Proceedings of the 1st International Conference on Pervasive Technologies Related to Assistive Environments*, page 30. ACM, 2008.
23. S. Schimke and C. Vielhauer. Similarity searching for on-line handwritten documents. *Journal on Multimodal User Interfaces*, 1(2):49–54, 2007.
24. T. Seidl and H. P. Kriegel. Efficient user-adaptable similarity search in large multimedia databases. In *VLDB '97: Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 506–515, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.
25. Ken Ueno, Xiaopeng Xi, Eamonn Keogh, and Dah-Jye Lee. Anytime classification using the nearest neighbor algorithm with applications to stream mining. In *ICDM*

'06: Proceedings of the Sixth International Conference on Data Mining, pages 623–632, Washington, DC, USA, 2006. IEEE Computer Society.
26. E. Vidal. An algorithm for finding nearest neighbours in (approximately) constant average time. *Pattern Recognition Letters*, 4(3):145–157, 1986.
27. Jules Vleugels and Remco Veltkamp. Efficient image retrieval through vantage objects. *Pattern Recognition*, 35:69–80, 1999.
28. R. A. Wagner and M. J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, 1974.
29. P. Wu and B. S. Manjunath. Adaptive nearest neighbor search for relevance feedback in large image databases. In *MULTIMEDIA'01: Proceedings of the ninth ACM international conference on Multimedia*, pages 89–97, New York, NY, USA, 2001. ACM.
30. P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search: The Metric Space Approach*. Springer, 2006.